***MOVIE MANAGEMENT SYSTEM***

***230962142- Akshay Dittakavi***

***230962095- Nandipati Vasistha***

***230962326- Karthikeya Davasam***

***230962150- Piyansu Bagaria***

***ABSTRACT:***

*The Movie Management System (MMS) is a database-driven solution designed to streamline movie records, categorize genres, manage screenings, and process ticket bookings. It ensures role-based access control, allowing administrators, staff, and users to interact with the system according to their roles.*

*Built on a relational database, the system ensures* ***efficient performance****. It allows users to* ***add, update, delete, and retrieve movie records****, schedule screenings, and book tickets. The system is designed to support enhancements like online booking and real-time ticket availability in the future.*

*This System includes 5 Relations (Tables)*

* *Movies Table*
* *Genres Table*
* *Screenings Table*
* *Users table*
* *Bookings Table*

**

***System Features***

***User Roles & Access Control***

* ***Admin:*** *Full control over movies, screenings, users, and bookings.*
* ***Staff:*** *Can manage screenings and process bookings.*
* ***Customers:*** *Can book movie tickets, view screenings, and track their bookings.*

***Movie Management***

* *Add, update, delete movie records.*
* *Categorize movies into genres.*
* *Track movie ratings and durations.*

***Screening & Booking Management***

* *Schedule movie screenings.*
* *Book tickets and manage available seats.*
* *Generate reports for business insights.*

***1️. Genres Table***

*Stores different movie genres (categories).*

***Attributes:***

* *genre\_id (NUMBER, PRIMARY KEY) → Unique ID for each genre.*
* *genre\_name (VARCHAR2(100), UNIQUE, NOT NULL) → Name of the genre (e.g., Action, Comedy).*

***Use:***

* *Categorizes movies for easy filtering and organization.*
* *Establishes a relationship with the* ***Movies*** *table.*

***2. Movies Table***

*Stores information about different movies.*

***Attributes:***

* *movie\_id (NUMBER, PRIMARY KEY) → Unique ID for each movie.*
* *title (VARCHAR2(255), NOT NULL) → Movie title (e.g., "Inception").*
* *genre\_id (NUMBER, FOREIGN KEY) → Links to* ***Genres*** *table to classify movies.*
* *release\_date (DATE) → Release date of the movie.*
* *duration (NUMBER, CHECK > 0) → Length of the movie in minutes.*
* *rating (NUMBER(3,1), CHECK 0-10) → IMDb-style rating (e.g., 8.8).*

***Use:***

* *Stores movie details like title, duration, and rating.*
* *Links to* ***Genres*** *via genre\_id to categorize movies.*
* *Used in* ***Screenings*** *to schedule movie showtimes.*

***3️. Screenings Table***

*Stores details of movie showtimes in different screening rooms.*

***Attributes:***

* *screening\_id (NUMBER, PRIMARY KEY) → Unique ID for each screening.*
* *movie\_id (NUMBER, FOREIGN KEY) → Links to the* ***Movies*** *table.*
* *screen\_number (NUMBER, NOT NULL) → The screen number where the movie is shown.*
* *show\_time (TIMESTAMP, NOT NULL) → The scheduled time of the screening.*
* *available\_seats (NUMBER, CHECK >= 0) → Number of seats available.*

***Use:***

* *Manages movie screenings by linking a movie to a specific screen and time.*
* *Keeps track of available seats for ticket booking.*
* *Used in* ***Bookings*** *for assigning customers to a screening.*

***4️. Users Table***

*Stores information about system users (admins, staff, customers).*

***Attributes:***

* *user\_id (NUMBER, PRIMARY KEY) → Unique ID for each user.*
* *username (VARCHAR2(100), UNIQUE, NOT NULL) → Login name for the user.*
* *password (VARCHAR2(255), NOT NULL) → Encrypted password for security.*
* *role (VARCHAR2(50), CHECK IN ('Admin', 'Staff', 'Customer')) → Defines user role.*

***Use:***

* *Manages user authentication and role-based access control.*
* *Admins can manage movies, screenings, and bookings.*
* *Staff can assist customers with bookings.*
* *Customers can book tickets.*

***5️. Bookings Table***

*Stores customer ticket reservations for screenings.*

***Attributes:***

* *booking\_id (NUMBER, PRIMARY KEY) → Unique ID for each booking.*
* *user\_id (NUMBER, FOREIGN KEY) → Links User table*
* *screening\_id (NUMBER, FOREIGN KEY)-> Links Screenings table*
* *seats\_booked (NUMBER, CHECK >0)*
* *booking\_date (DATE)*

***Use:***

* *Manages Bookings by user*
* *Connects Bookings table to User and Screenings table*

***CONCLUSION:***

*In summary, this mini project would successfully build a working Movie Management System using a database. It can store, find, change, and delete movie information like titles, genres, actors, and showtimes. This project shows how useful databases are for organizing information. While it's a basic version, it could be improved with features like online booking and user accounts. We learned a lot about database design and SQL programming while creating this system, which will be helpful in future projects.*